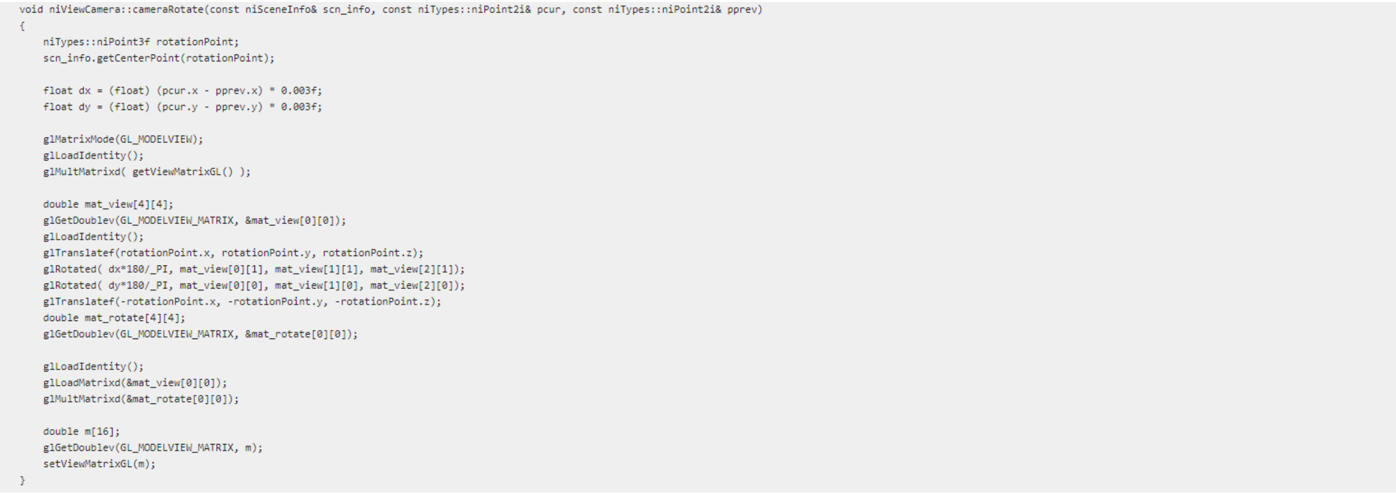
EXHIBIT C-2



```
void niViewCamera::cameraRotate(const niSceneInfo& scn_info, const niTypes::niPoint2i& pcur, const niTypes::niPoint2i& pprev)
{
    niTypes::niPoint3f rotationPoint;
    scn_info.getCenterPoint(rotationPoint);

    float dx = (float) (pcur.x - pprev.x) * 0.003f;
    float dy = (float) (pcur.y - pprev.y) * 0.003f;

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glMultMatrixd( getViewMatrixGL() );

    double mat_view[4][4];
    glGetDoublev(GL_MODELVIEW_MATRIX, &mat_view[0][0]);
    glLoadIdentity();
    glTranslatef(rotationPoint.x, rotationPoint.y, rotationPoint.z);
    glRotated( dx*180/_PI, mat_view[0][1], mat_view[1][1], mat_view[2][1]);
    glRotated( dy*180/_PI, mat_view[0][0], mat_view[1][0], mat_view[2][0]);
    glTranslatef(-rotationPoint.x, -rotationPoint.y, -rotationPoint.z);
    double mat_rotate[4][4];
    glGetDoublev(GL_MODELVIEW_MATRIX, &mat_rotate[0][0]);

    glLoadIdentity();
    glLoadMatrixd(&mat_view[0][0]);
    glMultMatrixd(&mat_rotate[0][0]);

    double m[16];
    glGetDoublev(GL_MODELVIEW_MATRIX, m);
    setViewMatrixGL(m);
}
```

Precession occurs here :

glRotated(dx*180/_PI, mat_view[0][1], mat_view[1][1], mat_view[2][1]);

glRotated(dy*180/_PI, mat_view[0][0], mat_view[1][0], mat_view[2][0]);

After the first rotation by dx, matrix view will change and become mat_view_dx, but in the code the next dy rotation is counted using to the old mat_view basis that was before the first rotation.